**Проект по „Структури от данни и програмиране“**

**Тема: “Музикална Колекция”**

Изготвил:

Кристина Бориславова Попова, ф.н. 81933, Компютърни науки, трети курс, втори поток, шеста група

# 1.Увод

## 1.1. Описание и идея на проекта

Основната идея на проекта “Музикална колекция“ е реализирането на музикална библиотека, която поддържа информация за песни и потребители и позволява създаването и премахването(на вече съществуващи) плейлисти според вкусовете на потребителя. Музикалната система съдържа голям брой функционалности, които улесняват потребителя.

## 1.2. Цел и задачи на разработката

Целта на проекта е създаването на музикална колекция, която да поддържа следните функционалности, с които позволява на потребителя да:

* влезне в сестемата с потребителско име и парола;
* направи регистрация с неизползвано потребителско име и парола;
* излезне от системата;
* промени своите лични данни, като потребителско име, парола, собствено име, дата на раждане;
* добавя и премахва жанр от своите любими;
* вижда кратка информация за поддържаните от програмата команди;
* добавя песни;
* създава плейлист под дадено име;
* премахва плейлист под дадено име;
* зарежда даден плейлист по име;
* вижда информация за всички песни в даден плейлист;
* гласува за дадена песен от музикалната библиотека;

# 2. Преглед на предметната област

## 2.1. Основни дефиниции и концепции , които ще бъдат използвани

# 3.Проектиране

## 3.1.Обща архитектура

За проекта „Музикална колекция “ бяха реализирани класовете User, Song, Playlist, Date, AlphabeticalSortedSongs и System. За работата с командите, беше използван поведенческият шаблон за дизайн - *command pattern,* за който бяха необходими следните класове :

* Invoker
* ICommand
* Validator – в който се валидира въведения вход от потребителя.
* AddFavGenre, AddPlaylist, AddSong, ChangeBirth, ChangeName,ChangePassword, ChangeUsername, FilterCommand, HelpCommand, LogInCommand, LogOutCommand, PrintUserPlaylist, RateCommand, RemoveFavGenre, RemovePlaylist, SignUpCommand

### 3.2. Диаграми

На фиг.1 са изобразени основните класове използвани в архитектурата на проекта “Музикална колекция”. Класът User се използва за представянето на потребителите в музикалната колекция. За всеки потребител се поддържа потребителско име и парола, пълно име, дата на раждане, множество любими жанрове и множество от плейлисти. Класът Song поддържа информация за името на песента и нейния изпълнител, жанр, албум, година на издаване и рейтинг. Класът Playlist представлява плейлистите, които всеки потребител създава. Всеки плейлист си има име и множество от песни, които съдържа.

В класът System се поддържат множество от потребители, множество от песни, които поддържа системата, текущ потребител, член данна, с която помним кой потребител вече е гласувал за дадена песен и флаг, който ни казва кога в системата има влезнал потребител.

Класът OrderedBinaryTree използваме за сортирането на песните по азбучен ред, когато викаме функцията filter().Използването на наредено двоично дърво ни позволява при всяко добавяве на нова песен да я поставим на правилното й място в дървото.Така всяка песен, която отговаря на исканите изисквания бива добавена в двоичното наредено дърво, като мястото й в него се определя от нейното име. Принтираме дървото чрез обхождането ляво-корен-дясно, като по този начин принтираме песните по азбучен ред. Клъсът е така реализиран, че ако вече дадена песен се намира в двоичното дърво, тя да не се добави.

Shape, polygon

Description automatically generated

**фиг.1**

# 4.Реализация и тестване

## 4.1. Реализация на класове

Класът Date използваме за датата на раждане на потребитела и датата, на която дадена песен е издадена. По-важните членфункции реализирани в този клас са:

* bool operator==(const Date& other)const; - която връща

истина, ако две дати са едни и същи, и лъжа, когато са различни. Тази функционалност е необходима за да можем да проверяваме дали две представления се провеждат на една и съща дата.

* int getDay()const; - *гетър* връщащ деня.
* int getMonth()const; - *гетър* връщащ месеца. • int getYear()const; - *гетър* връщащ годината.

Класът User потребителят, който използва системата. В него бяха реализирани *гетъри,* които ни връщат член данните на класа. Член функции, с които да можем да променяме(*сетъри*) член данните на класа. Останалите член функциите са:

* void removeFavGenre(const std::string& genre);- с която премахваме любим жанр;
* void addFavGenre(const std::string& genre); - с която добавяме любим жанр;
* bool createPlaylist(Playlist\* playlist); - с която създаваме нов плейлист;
* void removePlaylist(Playlist\* playlist); - с която премахваме даден плейлист;
* void addSongToPlaylist(const std::string& pl, Song\* song); - с който в даден плейлист добавяме нова песен;

Класът Song e пeсента. В него бяха реализирани *гетъри,* които ни връщат член данните на класа и функцията void rate(int n);, която използваме, когато даден потребител гласува за дадена песен.

Клъсът Playlist прадставлява плейлиста, който всеки потребител създава и добавя песни в него. В него имаме два *гетъра* и член функцията void addSong(Song\* song);, с която добавяме песни в плейлиста.

В класът System се прочитат всички потребители, които имат вече създадени профили в системата, зареждат се всички песни, които поддържа музикалната колекция, помни се текущият влезнал потребител и дали той е гласувал за определена песен.Член-функциите реализирани в този клас са всички операции, които трябва да поддържа музикалната система.

* void signup(const std::string& username, const std::string& password, const std::string& name, int day, int month,int year);
* void login(const std::string& username, const std::string& password);
* void logout();
* void changeUsername(const std::string& username);
* void changePassword(const std::string& password);
* void changeDateOfBirth(int day, int month, int year);
* void changeFullName(const std::string& fullName);
* void addFavGenre(const std::string& genre);
* void removeFavGenre(const std::string& genre);
* void addSong(const std::string& playlist, const std::string& name, const std::string& artist, const std::string& genre, const std::string& album, int day, int month, int year);
* bool addPlaylist(const std::string& playlist);
* void removePlaylist(const std::string& playlist);
* void rateSong(const std::string& name,int rate);
* void filter(const std::string& input, const std::string& playlistName);
* void help()const;
* void printUserPlaylist(const std::string& playlist);

**4.2. Създаване на тестови сценарии (създаване на примери)** Системата беше тествана върху следните тестови сценарии:

# 5.Заключение

## 5.1. Обобщение на изпълнението на началните цели

Проектът удовлетворява исканите функционалности. Бяха създадени, разработени и разширени (с допълнителни член-функции и член-данни) исканите класове. Създадоха се и допълнителни класове, с които да се изпълнят поставените задачи. Системата беше тествана и коригирана, когато не отговаряше на изискванията.

**5.2. Насоки за бъдещо развитие и усъвършенстване**

Нещата, които биха усъвършенствали проекта са:

* създаване на графичен интерфейс за по-удобно използване на системата.

# Използвана литература

1. Тодорова, Магдалина, “Обектно-ориентирано програмиране на базата на езика C++”,